Blackboardlearn+

Release 9.1

Snapshot Command Line Tool

WARNING! The Snapshot Command Line Tool is deprecated in Blackboard Learn 9.1 Service Pack 8 (SP8). Documentation and support will discontinue in future releases. To create new data integrations, see Student Information System Integrations in Help.Blackboard.Com.

Blackboard

Publication Date: December 28, 2011 Revision: 0

Worldwide Headquarters	International Headquarters
Blackboard Inc.	**Blackboard International B.V.**
650 Massachusetts Avenue NW Sixth Floor Washington, DC 20001-3796	Paleisstraat 1-5 1012RB Amsterdam The Netherlands
+1 800 424 9299 toll free US & Canada	
+1 202 463 4860 telephone	+31 (0) 20 788 2450 (NL) telephone
+1 202 463 4863 facsimile	+31 (0) 20 788 2451 (NL) facsimile
www.blackboard.com	www.blackboard.com

Contents

Introduction

Populating and synchronizing the Blackboard Learn database with data from an institution's information systems is a continual process that can be accomplished in many ways. Blackboard Learn is installed with the tools to assist with this process, but every school is unique and has different requirements. To establish a successful process, careful planning is necessary as well as an understanding of the underlying systems involved.

This guide provides background information about integration and explains the standards of Blackboard Learn. It also provides suggestions for institutional planning as well as specific instructions and resources for using the integration tools included with Blackboard Learn.

The integration tools that are included with Blackboard Learn can be used to integrate Course data and Organization data. The text of this document references both Course and Organization data; however, Organizations are only enabled with the licensing of Community Engagement.

Data integration

Data integration is the efficient and timely automatic updating of data from an institutional system to Blackboard Learn. By creating an automatic process, institutions save time and resources as well as build flexibility into the system by avoiding the manual transfer of user, course, and enrollment data into Blackboard Learn.

To create this automated process, planning and implementation must occur. Blackboard Client Support and Blackboard Consulting can provide the necessary support to institutions for successful data integration. Blackboard Client Support can assist with core Blackboard questions and issues. They can be contacted at http://behind.blackboard.com. Blackboard Consulting can assist with strategy, planning and implementation of products or key features, customizations, and integration with other systems.

In This Guide

This guide is divided into several parts: The first part examines the principles of data integration including strategic planning, the second part explains how to manage different data sources, and the third part explains the Snapshot Commend Line tool and how to use it to populate the Blackboard Learn database.

For data elements and sample files see Help.Blackboard.Com. To access this information, you will need a login.

Important Note: This guide will not be updated. Snapshot is a deprecated process as of the release of Blackboard Learn Service Pack 8. Please see Data Integration in help.blackboard.com for the recommended methods of integrating Student Information Systems data into Blackboard.

Strategic Planning for Data Integration

This section describes the relationship between institutional data and Blackboard Learn data, the data integration process, and the methods of integrating institutional data with Blackboard Learn. Emphasis is placed on strategic planning to map the types of institutional data to be added into Blackboard Learn, when it will be added, and how the data is imported.

The data integration process involves the planning and design of the systems used to transfer user, course, enrollment, and staff data between an institution's information systems and Blackboard Learn. The integration process is a collaborative effort between Blackboard Learn and the institution. Blackboard Consulting can provide the support, experience, and knowledge necessary to assist institutions with an efficient integration.

Importing user, course, enrollment, and staff data from an institution's information systems into Blackboard Learn can involve many different systems and different people. Blackboard Learn is installed with the tools to assist with this process, but every organization is unique and has different requirements.

Questions to Ask Prior to Integration

Institutions must answer the following questions prior to data integration:

- Which institutional systems contain the data for integration with Blackboard Learn?
- How will attributes of the institutional system map to Blackboard Learn?
- Which criteria must be satisfied before a data record can be sent from the institutional system to Blackboard Learn?
- How will integration provide the needed packaging and error handling practices required by the institution?

Project Teams

To successfully plan and implement a data integration project, Blackboard recommends forming cross-functional teams representing each stakeholder in the project. These teams usually consist of:

- An institutional Project Manager
- Analysts with detailed understanding of the institution's business processes, needs, and associated information systems
- Programmers with the ability to query, report from, and interface with the enterprise systems

If you are working with Blackboard Consulting on the data integration project, these teams should also represent Blackboard, the institution, and solution partners. These teams may consist of:

- An institutional Project Manager
- A Blackboard Project Manager
- Institutional analysts with detailed understanding of the institution's business processes, needs, and associated information systems
- Institutional programmers with the ability to query, report from, and interface with the enterprise systems
- Blackboard and solution-partner technical consultants

Steps to a Successful Integration

Once your teams have been formed, you will need to put together a project plan. We have identified these seven steps to take for a successful integration.

1. Define the scope and time frame of the data integration project:
    a. What data entities will be exchanged between the systems?
    b. When will the project be complete?
2. Develop, analyze, and document requirements, including:
    a. Ownership of data entities and attributes
    b. Key definition
    c. Attribute mapping
    d. Data modeling and trigger determination
    e. Timeliness
    f. Reliability
    g. Security
    h. Operational issues.
3. Design, configure, and program a data integration solution. Blackboard Learn features many built-in integration capabilities, but some institutions may require additional customization.
    a. The project team can use existing solutions from enterprise vendors
    b. Reuse prior solutions developed for other institutions
    c. Develop an entirely new custom solution
4. Develop and document operating procedures for all affected parties.
5. Test the system.
6. Transfer the technology to the operations team at the institution.
7. Train all relevant personnel.

The Data Intergation Planning Process

Document the planning, design, configuration, and programming of the data integration process of steps two and three in the previous list. The planning portion consists of developing the following:

- **Data mapping**: organize the data sources to prepare the data dictionary (a document that lists the various acceptable attributes for each data field) and map required and optional attributes of Blackboard Learn to those of institutional systems. The Blackboard Data Dictionary begins with Category Data Feed Elements.
- **Data lifecycle**: generate business rules that regulate how data is handled, for example when records are added, removed or archived.
- **Data operations**: generate rules that govern the use of the integration tools to transfer data, for example how the system is initially populated and how it is synchronized with other systems.

Tips for Success

Complete the data integration for one information system at a time. It is quicker to add a second auxiliary system after the first is done.

When planning the integration, focus on getting the data into the system before determining when to remove it. In most cases, institutions will have a whole semester or longer to complete the rules for disabling data so the plan to remove it does not have to be completed up front.

For assistance in planning data integration, please contact Blackboard Consulting.

Data Entities and Standardizing Information Models

A data entity is a type or group of data records that share the same attributes. An easy way to visualize a data entity is to view it as a table, with attributes as columns and records as rows. A unique identifier called a primary key, or a key, defines each record in an entity.

For systems to be able to exchange data, standardized structures need to be in place. The Instructional Management System (IMS) consortium defines the information model used by Blackboard Learn. IMS is a global effort whose primary objective is to define a standardized set of structures that can be used to exchange data between different instruction-based systems. IMS has defined an information model and provided a specification for transfer of such data.

To learn more, visit the IMS Web site: http://www.imsproject.org/.

Blackboard Learn data entities adhere to the IMS standards for data, making it possible to consistently map data entities from a variety of sources to data entities in the Blackboard platform.

Data Entities

Data that can be input to Blackboard Learn is grouped into the following eleven entities:

- **User**: Data consists of system role, ID, contact, and personal information.
- **Course**: Data consists of Course Name, course section ID, and description information.
- **Organization**: Data consists of Organization Name, organization section ID, and description information.
- **Student**: Data consists of the core information required to assign a student to a given course section.
- **Staff Student**: Data consists of the core information required to assign a staff member to a given course section.
- **Organization Membership**: Data consists of the core information required to establish a user's membership in a given organization.
- **Course Category**: Data consists of the title, key, and availability information of a Course Catalog category.
- **Course Category Membership**: Data consists of the information required to connect a course to a category.
- **Organization Category**: Data consists of the title, key, and availability information of an Organization Catalog category.
- **Organization Category Membership**: Data consists of the information required to connect an organization to a category.
- **User Institution Role Membership**: Data consists of the information required to connect a user with multiple Institution Roles.

User ID

When updating the user ID of a Blackboard Learn user, the file permissions that are applied to a course or group membership or role are not impacted by this change. When the user ID is changed, Web Folder/Shared Location links that pointed to the user ID will have to be updated to use the new user ID information.

Note: On the Mac, a Web Folder is called a Shared Location.

Standardized Structures

The structures within Blackboard Learn that are standardized to use the IMS model for exchanging data allow for the successful mapping of an institution's data into Blackboard Learn. The following table details these structures.

Blackboard Learn Data Entity	IMS Data Entity	Description
User	Person	An end user of Blackboard Learn, including students, staff, and others.
Course	Group	An instance of an online class delivered through the Blackboard Learn, typically a course section.
Organization	Group	A site for groups, Organizations, clubs, and teams to have an online presence, share information, engage in increased communication, and collaborate on tasks.
Student enrollment	Group membership	The record establishing a student's participation in a course website.
Staff Student	Group membership	The record establishing a staff member's participation in a course website and their role in the delivery of the course.
Organization membership	Group membership	The record establishing a user's membership in an Organization and their role in the organization.

Data Life Cycle Planning

Data life cycle planning defines the rules that control how your institution manages data. These rules will determine what data is transferred to Blackboard Learn, when it is transferred, as well as any other conditions for data handling.

Business Rules Drive Data Integration

Data life cycle planning involves the creation of business rules. Business rules are statements that define or constrain some aspect of the business. They assert structure to control or influence the behavior of the business. The business rules set by your school controls whether data may be created or changed, when data can be sent, and the implications when this occurs.

Some important business rules to consider are:

- Where data is stored?
- What business processes impact the data?
- Which records to send?
- How often to send records?
- When to stop sending specific records?
- When to archive course website and organization website records?

Documenting the business rules of an institution will provide the foundation for what data is integrated by which methods over time. Once documented, these rules can be applied to the systems involved ensuring that data from each institutional system is integrated into Blackboard Learn exactly as intended.

The Data Life Cycle and Triggering Events

To plan for the successful transfer of data from institutional systems into Blackboard Learn, the life cycle of data has to be taken into consideration. The data life cycle is the series of events that control the actions of a record from its creation to when it is deleted. A data record in the Blackboard platform has a life cycle of usefulness and an associated status that it can pass through as its usefulness changes. The system processes the data differently based on its state in the life cycle.

After creation, data can exist in two states within Blackboard Learn; it can be active (enabled) or it can be disabled. Disabling a record rather than deleting it is often the best option if the record will be activated at a later time. For example, a student record can be disabled for a semester if that student is not taking any online courses, but may do so in the future. Deleting a record removes it from Blackboard Learn altogether, and if the student in the example were deleted, the record would have to be recreated and added back into Blackboard if that student decided to take another online course the following year.

All data entities are disabled using data integration tools. Course and organizations are the only data entities that may be disabled through the user interface. They can be introduced into the system at a later date through the same user interface.

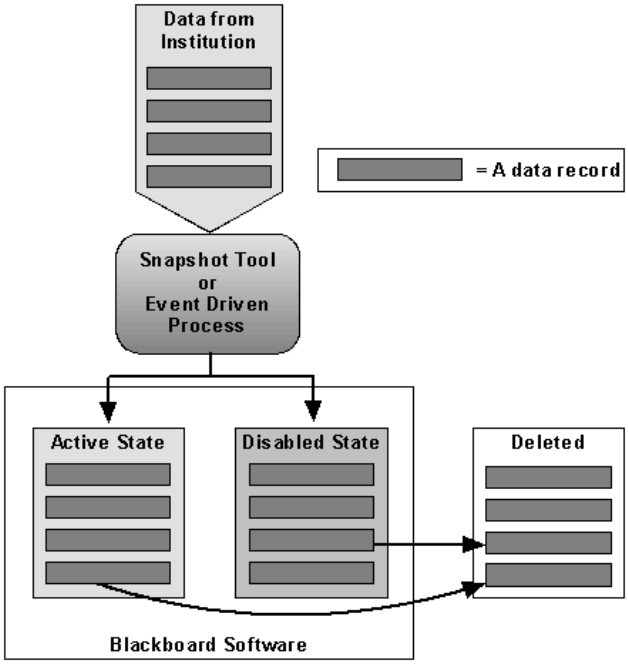
Triggering Events

A triggering event is an event that impacts (adds to, updates, or deletes) data within the Blackboard platform. When planning data integration, written definitions of triggering events and complete documentation of the criteria for adding, updating, disabling, or deleting data will increase the long-term integrity of the system and promote organizational consistency.

For example, an institution can define a triggering event as a student adding or dropping a course within a defined timeframe. If the student adds a course within the defined timeframe, the system can trigger an event to add that student record to the course. If a student drops a course within the defined timeframe, the system can trigger an event to disable that student record for the course. If the student drops a course outside of the defined timeframe, the system will not trigger an event and the student record will not be disabled.

Data Life Cycle

Each data record has a lifecycle. A data life cycle is the creation, update, and archiving or deletion of a record over time.



Data States

All records within the system exist in one of three states: active, disabled, or deleted. The status of that state is changed based on the life cycle of the data in the system. A record with a status of active can be disabled or deleted. A disabled record can be activated or deleted.

Disabling active objects and enabling inactive objects is done through the Snapshot Command Line Tool or the Event Driven API. Deleting a record can be accomplished through the integration tools or through the Administrator Panel in the user interface. Course and organization content are the only data entities that can be archived before being deleted. Archived data can be downloaded from and restored to the Blackboard Learn platform through the user interface or using batch tools.

In addition to the state, most record types have an additional property called “availability,” which directly correlates to the record's usage. The availability property can be controlled through the integration tools (Snapshot or Event-Driven API) and through the Administrator Panel in the user interface.

Behavior of Data Records in Relation to Their State

The following table details each entity and a description of its records' behavior in certain states within Blackboard Learn.

Entity	State	Behavior in Blackboard Learn
User	Active	Default state. The user record appears in the system.
Disabled	The user record exists within the database but is not active within the system. The user cannot login, however the user record appears in the System Control Panel in gray text with an icon that signals its disabled status.	
Deleted	Deletes user data and Course and organization membership.	
Course or Organization	Active	Default state. The Course appears within Blackboard Learn.
Disabled	The Course does not appear to any users within Blackboard Learn user interface; however it still exists in the database. The record appears in the System Control Panel in gray text with an icon that signals its disabled status.	
Deleted	Deletes all Course content, enrollments, and all activity data	

Entity	State	Behavior in Blackboard Learn
	associated with the Course.	
Student Enrollment or Organization Membership Student Enrollment or Organization Membership	Active	Default state. The enrollment appears within Blackboard Learn.
Disabled	The Student enrollment record is in the database but cannot be viewed through the user interface except by the Administrator. The record appears in the System Control Panel in gray text with a circle icon with a line through it to signal it is disabled.	
Deleted	The enrollment record is deleted from Blackboard Learn and all activity associated with the enrollment with the exception of Discussion Board messages.	
Staff Student	Active	Default state. The Student appears within Blackboard Learn.
Disabled	The Student record appears in the database but not in the user interface. The Instructor or staff member may not access the Course.	
Deleted	The record is deleted from Blackboard Learn.	
Course or Organization Category	Active	Default state. The category appears within Blackboard Learn.
Disabled	The category exists in the database but does not appear in Blackboard Learn. Courses listed under the category are not deleted.	
Deleted	The category does not exist in Blackboard Learn.	
Course or Organization Category Membership	Active	Default state. The link appears within Blackboard Learn.
Disabled	The link exists in the database but does not appear in Blackboard Learn.	
Deleted	The link does not exist in Blackboard Learn.	
User Institution Role Membership	Active	Default state. The user is able to access the appropriate privileges of the

Entity	State	Behavior in Blackboard Learn
		Institution Role.
Disabled	The link exists in the database the user does not have the privileges of the Institution Role.	
Deleted	The link does not exist in Blackboard Learn.	

Behavior of Records in Active State

The following table details the behavior of active entities within Blackboard Learn based on availability.

Entity	Availability	Behavior in Blackboard Learn
User	Available	Default. Can access features and functions of Blackboard as defined by roles.
Unavailable	User cannot log in to the system. Instructors can see the record in course and organization interfaces. The User Name appears as gray in the user interface with a circle icon with a line through it that signals it is unavailable.	
Course or Organization	Available	Default. Can be accessed by all authorized users.
Unavailable	Only administrators and course staff can view and access the course. Does not affect enrollments and staff Students.	
Student Enrollment or Organization Membership	Available	Default. Enables users to access their Courses or organizations.
Unavailable	Instructors and administrators can see the unavailable Student enrollment within the course/organization membership list. The record appears as gray on the user interface.	
Staff Student	Available	Default. Enables staff to access their Courses and organizations.
Unavailable	Staff and administrators can see the unavailable Student within the course membership list. The record appears in gray and the course is inaccessible to the staff member.	

Data Operations Planning

Data Operations planning involves choosing the data integration tool or tools that best suits the needs of the institution and applying the business rules developed during the data lifecycle planning.

Importing Data into the Blackboard Learn Database

Blackboard Learn imports administrative data using a data feed, which can be done using a flat file, an XML file, or the Event Driven API (Application Program Interface). A data feed request can:

- Add a new record
- Update an existing record
- Delete an existing record
- Change the key of a record

WARNING! Take care when changing the key of a record. Changing the key or unique identifier of a record should be done sparingly and with the greatest care. However, there are some situations where it is necessary. For example, if Social Security numbers have been used as the key for user records and for security and privacy purposes student numbers are being issued to replace social security numbers, it makes sense to change the key within the system.

Methods of Integration

There are two basic methods of data integration: using the Snapshot Command Line Tool and the Event Driven API. The two methods can be combined into a composite method, configured according to your school specifications, to use the strengths of each method while off-setting each method's limitations.

Snapshot-based Approach

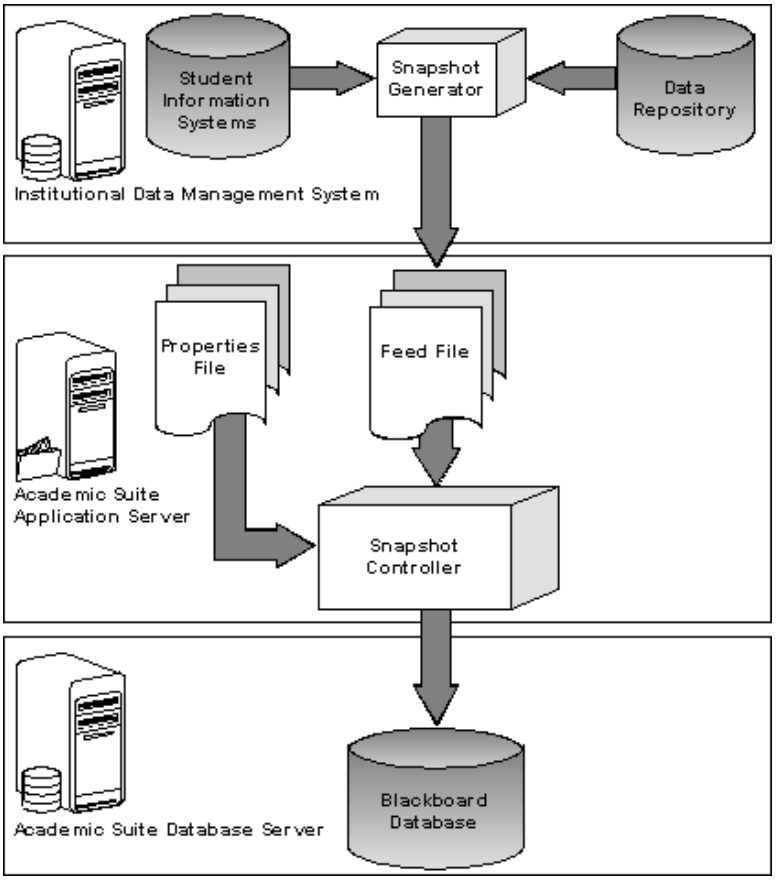
Snapshot allows all data for an entity to be transferred from an institutional information system to the Blackboard Learn database at a particular point in time. It is called a snapshot because it is akin to taking a picture of the institutional database at that point in time and transferring the contents to Blackboard Learn database.

In a snapshot-based approach, snapshots are taken at pre-determined times by the Snapshot Generator. The Snapshot Generator is a utility usually written by your school. This utility extracts data from the Student Information System (SIS) or other database external to Blackboard Learn, and formats it for input into the Snapshot process, one file per type per data source. In automated environments, it is used in conjunction with the Snapshot Controller.

The Snapshot Controller is a utility that is written to manage daily data feeds created by a "snapshot generator" and feed them to the Snapshot tool. The controller automates

and manages the process from the time a data feed leaves the Snapshot Generator until it reaches the Blackboard Learn database. The Snapshot Controller is usually provided by Blackboard Consulting. It can be customized for each institution so that it can direct email to appropriate people if a failure occurs in the process, making failure handling relatively simple during the Snapshot process.

Note: The Snapshot Controller is not necessary for Data Integration. The Snapshot Controller is not included with the Blackboard Learn and may be written by your school.



The Snapshot-based approach offers higher failure tolerance than the Data Integration approach, but reduces synchronization because reports of relevant data (“snapshots”) are run at regularly scheduled times, for example once a day, not dynamically. Snapshot is best used to handle large batches of data. For example, the initial population of Blackboard Learn database is easily accomplished using the Snapshot-based solution.

The Snapshot Tool sends commands to the Blackboard Learn database instructing it to update any data that has changed since the last comparison. In the event of a Blackboard Learn database failure, Administrators can run another snapshot or process the last snapshot. Because the entire data set must be compared each time the Snapshot Tool is run, this approach's performance is slower than the Event-Driven Integration approach. Dividing snapshots into logical subsets, using Data Source Keys, can alleviate some of the problems associated with a bulky Snapshot process.

Data sources allow for:

- Smaller files
- Faster processing
- Different update patterns

To learn more about identifying groups of data and constructing data sets and Data Source Keys, see About Data Entities.

When the Snapshot Tool is used in high-demand environments where certain information in Blackboard Learn must be routinely updated on a day-to-day basis, the ability to use Snapshot manually or with basic scripting may become too tedious or difficult to manage efficiently and effectively. In these circumstances, Blackboard can assist in the creation of custom scripted Snapshot routines that integrate Blackboard with a Student Information System or build this integration from scratch with integration assistance from your technical staff. For more detailed information, contact Blackboard Consulting.

Event Driven APIs Approach

The Event Driven API allows institutional systems and Blackboard Learn database to exchange event information in real time. When an event occurs (add, update, or delete) that changes the state of the data in the Institution systems, a transaction is sent to Blackboard Learn database to provide synchronization between the two systems dynamically.

Under the Event-Driven approach, when a triggering event takes place, a command is automatically sent to Blackboard Learn database. Blackboard Learn database receives the command, executes it, and returns a status update to the information system. If a failure occurs, the command is saved or the event is "marked" until a backup takes place and the command can be re-sent successfully. To implement this approach, access to triggers in the enterprise system is required.

Combined Snapshot and Event Driven APIs Approach

The combined approach attempts to use the best aspects of both methods. First, the Snapshot approach is used to achieve improved failure tolerance at the expense of performance. Then, the important events that require quick synchronization with Blackboard Learn database are handled using the Event Driven API approach. Using this approach, an Institution can reduce the complexity of managing failures while

maintaining a high level of synchronization. It can be implemented for all triggering events or just the most significant.

Data Source Management

Data Source Management (DSM) is a flexible method for identifying sets of data from multiple external sources so that they can be managed within a single Blackboard Learn database. A data source is a logical set of data, sometimes simply associated with a source system, other times associated with type and term data as well.

During the planning process, your school's business rules were identified as well as all the systems involved with storing institutional data that will be used in Blackboard Learn to create courses, populate courses, disable and delete records, and so on. This information is vital to deciding how to organize and manipulate the data within the Blackboard Learn database.

All data pushed to the Blackboard Learn database is tagged with a data source identifier. These identifiers, called Data Source Keys (DSKs), are created using the Data Source Management (DSM) Tool and are the means by which certain data sets are managed. As of Blackboard Learn Service Pack 4, Data Source Keys can be created from the user interface on the Administrator Panel.

Data Source Keys parse large amounts of data into smaller entities by tagging them with a unique identifier. This identifier ties pieces of data together to ensure that they are processed within the system according to institutional practices and identified business rules.

For example, a school has courses that are given in the fall semester, the spring semester and in the summer. All the records come from the same external data source, a database in the Office of the Registrar. If all the courses were kept in the same data set when importing this data into Blackboard Learn, everything time a change needed to take place, such as removing the fall courses from the active course catalog, all the courses would be affected because there is no way to differentiate the fall courses from the spring or summer courses even though they have different attributes. By assigning Data Source Keys to each of the three time frames, courses that are assigned to the fall can be manipulated without affecting the spring or summer courses.

About Data Source Keys (DSKs)

Data Source Keys are labels made up of alpha-numeric strings that allow different types of data from a single data source to be grouped together so that they can be handled in a single operation. The Snapshot Command Line Tool and Snapshot feed files used for SIS integration process data based on what Data Source Key it is associated with. Using Data Source Keys breaks up Snapshot processes into separate operations to optimize system resources and meet business rules.

Data Source Keys are created as needed and can be saved for future use. They can be used in almost infinite ways to categorize data, tying together data from different systems so it can be added to Blackboard Learn. Data Source Keys are stored in the Blackboard database and are referenced in the Snapshot properties file.

Example of a Data Source Keys

ACME University teaches 1,000 courses twice a year (Fall and Spring). Rather than lump all those courses together, it would be easier to process them if they could be separated by the semester in which they occur. By associating the courses in each semester with two different Data Source Keys, one for Fall and one for Spring, the courses in each semester can be manipulated separately.

For instance, ACME assigns the `FALL_Courses_11` Data Source Key to all fall courses and the `SPRING_COURSES_12` Data Source Key to all spring courses. This makes processing them much easier by keeping them separate. If these courses were not identified with separate Data Source Keys, then it would be difficult to add, change, or remove all Fall courses from the system without also modifying the Spring courses as well.

Data Source Keys can be used in many ways to categorize data but there are some general tips that should be followed when creating and applying Data Source Keys to data.

- Keep a consistent naming convention for the Data Source Keys to prevent confusion when it is time to modify or remove data.
- Avoid creating multiple Data Source Keys for entries that will remain for a long period of time (such as Students or Faculty). Doing so may create unnecessary complications or problems.
- When archiving and removing courses at the end of a semester, it is best to disable them first for a short period of time before archiving and removing them from the system. This will allow a short period to confirm that they have been archived safely before removing them and help prevent accidental deletion of courses that have not been safely preserved if desired.

Note: When assigning data sources to course and organization categories, child categories must belong to the same data source as the parent category when the category tree is inserted. If child categories do not appear in the same data source as the parent, the child-parent relationship will not be maintained.

Sample Naming Conventions for Data Source Keys

In order to ensure logical application and knowledge transfer, create a system for naming Source Keys so that they can be easily identified. The following naming convention represents a relatively simple way to break up the data sets to enable the two most common workflows.

Data Source ID

A simple ID should be assigned for the source system, for example SIS for a Student information system, or HRMS for a human resources management system. By

combining this ID with an ID for each type of set, a flexible naming scheme can be derived to support typical workflows.

Type Bound Sets

Type bound sets include a component that is derived from the type of feed that is being performed. This way a data set is identified by the entities involved. For example, if the string 'course' is used to mean "course" then that string is included to indicate the type of the data set, for example `SIS.COURSE`.

Term Bound Sets

Term bound sets are used to group data that is related, but should not overlap time periods in the database. For example, it might be desirable to feed spring courses into the database while fall courses are still active. Using a key that distinguishes the two sets based on their term will prevent Snapshot Command Line operations on one set from interfering with data in the other. For example, `SIS.SPRING2012` and `SIS.FALL2011`.

Type and Term Bound Sets

In many cases it is desirable to use a combination of type and term bound identification. The most common example is Student enrollment at an institution with a fixed academic calendar—enrollment is bound to a specific semester for example, `SIS.COURSE.FALL2011`.

Example

A school wants to process students, Instructor lists, course section listings, and enrollments over the course of several semesters. In general, from semester to semester, the student and instructor lists will encompass the same basic data set. However the courses and enrollments will need to be processed on a per-semester basis. That is, from semester to semester, active students and staff will be treated as a single logical set (with fluid membership) while courses and enrollments will be treated as logically distinct sets that do not intersect from semester to semester.

One solution is to use type-bound keys for students and instructors, and type and term bound keys for courses and enrollment. A Data Source Key is created called SIS.USERS that is used to identify the set of users over time. By running Snapshot Command Line feeds on all ongoing users, all active students, and staff may be processed as a single set of data.

Separate Data Source Keys are created for courses, enrollment, and instructor students, all which are both type and term bound:

`SIS.COURSE.FALL2011`

`SIS.COURSE.SPRING2012`

That way, all user feeds may use the `SIS.USERS` Data Source Key, while courses and enrollment can use the `SIS.COURSE.*` keys, with guarantees that operations performed on one of the sets will not affect the other set.

As another example, different sets can be applied to different users:

```
MEDSIS.USERS
SIS.USERS
ALUMNI.USERS
```

Illegal characters for Data Source Names

Data source keys should consist only of the letters A-Z, numbers 0-9, periods, and underscores (_).

Data Source Key names are not case sensitive.

How to Create Data Source Keys

From the Administration Panel

1. On the **Administration Panel**, in the **Building Blocks** section, click **Data Integration**.
2. Click **Data Sources**.
3. Click **Create Data Source**.
4. Type a unique **Key** and optionally add a **Description**.
5. Click **Submit**.

Using the Data Source Management (DSM) Command Line Tool

Data Source Keys can also be created using the Data Source Management (DSM) tool. The DSM tool is a command line application that creates and manages Data Source Keys and associates them with their respective data entities. Once created by the DSM Tool, Data Source Keys are stored in the Blackboard database. Data Source Keys have no meaning, but become another attribute of the data that they are associated with.

The DSM tool can be used with Snapshot and the Event Driven APIs.

The DSM tool is installed when Blackboard Learn is installed. The DSM tool is located in `blackboard_home\apps\snapshot\bin\` in a typical installation.

The DSM Tool

Located in the `blackboard\apps\snapshot\bin\` directory in a typical installation is a `readme.txt` file that lists all the available commands for both the Data Source Management (DMS) tool and the Snapshot Command Line tool. The entire `readme.txt` file is located in Sample Properties File.

DSM Tool Syntax

Commands issued to the DSM Tool must follow a specific syntax in order to be processed correctly. Flags are used to specify arguments in a DSM Tool command.

DSM Tool Commands support the following flags:

- The “-f” flag and the invocation (command) that is to be run
- The “-b” flag and the Data Source Key that is to be used
- The “-V” flag and the fully-qualified domain name of the virtual host / server name that will be used

- The “-d” flag and the description of the Data Source Key (the description must be enclosed with quotes)*
- The “-r” flag and the replacement Data Source Key
- The “-t” flag and the date*

Flags are case sensitive, so an upper case “V” needs to be used to denote the virtual host name, and the other flags are in lower case.

Flags with an asterisk (*) are optional and can be omitted.

Example: Windows

```
DSM –f [action] –b [data source key] –V [fully-qualified server name]
```

Example: Unix

```
./dsm.sh –f [action] –b [data source key] –V [fully-qualified server name]
```

Taking the above criteria, a sample DSM command formatted in the correct syntax can be built like this:

```
DSM –f REMOVE –b Fall_COURSES_11 –V blackboardserver.acme.edu
```

Note: The DSM tool’s parameters are not operating system specific. The only difference is that UNIX systems use the `./dsm.sh` invocation while Windows systems use simply `dsm` or `dsm.cmd`.

DSM Tool Commands

To create and manage Data Source Keys, use the DSM Tool commands listed in the following table with the DSM Tool Syntax.

DSM Tool Command	Description
LIST	Lists all data sources.
CREATE	To create a new data source.
REMOVE	To remove a data source.
MODIFY	To modify a data source. The modify command can be used to change the description, the key, or both.
COUNTS_ALL	To count all records for a data source. The count includes both enabled and disabled records.
COUNTS_DISABLED	To list the count of disabled records for a data source.
COUNTS_ENABLED	To list the count of enabled records for a data source.
DISABLE_ALL	To disable all entities for a data source.

DSM Tool Command	Description
DISABLE_PERSON	To disable all person (user) entities for a data source.
DISABLE_GROUP	To disable all group (Course/Organization) entities for a data source.
DISABLE_MEMBERSHIP	To disable all membership (enrollment / staff Student) entities for a data source.
DISABLE_COURSE_CATEGORIES	To disable all course category entities for a data source. Note that all category memberships related to this category are not disabled in the database but are rendered unusable due to the disabled category.
DISABLE_ORGANIZATION_CATEGORIES	To disable all organization category entities for a data source. Note that all category memberships related to this category are not disabled in the database but are rendered unusable due to the disabled category.
DISABLE_COURSE_CATEGORY_MEMBERSHIP	To disable all course category memberships for a data source.
DISABLE_ORGANIZATION_CATEGORY_MEMBERSHIP	To disable all organization category memberships for a data source.
DISABLE_INSTITUTION_ROLE_MEMBERSHIP	To disable all Institution Role memberships for a data source.
PURGE_ALL	To purge all entities for a data source. Purge only deletes disabled items.
PURGE_PERSON	To purge all person (user) entities for a data source. Purge only deletes disabled items.
PURGE_GROUP	To purge all group (Course/Organization) entities for a data source. Purge only deletes disabled items.
PURGE_MEMBERSHIP	To purge all membership (enrollment / staff Student) entities for a data source. Purge only deletes disabled items.
PURGE_COURSE_CATEGORIES	To purge all course category entities for a data source. This will also purge all category memberships related to the category. Purge only deletes disabled items.
PURGE_ORGANIZATION_CATEGORIES	To purge all organization category entities for a data source. This will also purge all category memberships related to the category. Purge only deletes disabled items.
PURGE_COURSE_CATEGORY_MEMBERSHIP	To purge all course category memberships for a data source. Purge only deletes disabled items.

DSM Tool Command	Description
PURGE_ORGANIZATION_CATEGORY_MEMBERSHIP	To purge all Organization category memberships for a data source. Purge only deletes disabled items.
PURGE_INSTITUTION_ROLE_MEMBERSHIP	To purge all Institution Role membership entities for a data source. Purge only deletes disabled items.
PURGE_SESSION	To clear all persistent storage devices used by Snapshot. Use this when Snapshot and COPYINTO operations are slow. This clears temp table data and cached data.

About the Snapshot Command Line Tool

Snapshot is a simple command line program that takes a feed file and pushes the data into the Blackboard Learn database. It is built for bulk operations and has several modes that can be used to facilitate different workflows. Snapshot is automatically installed as part of Blackboard Learn. It can also be installed on a separate computer, which may be necessary for clients using Blackboard ASP, or for certain network setups.

Snapshot allows all data for an entity to be transferred from an institutional information system to the Blackboard database at a particular point in time. It is called a “snapshot” because it is akin to taking a picture of the institutional database at that point in time and transferring the contents to Blackboard Learn.

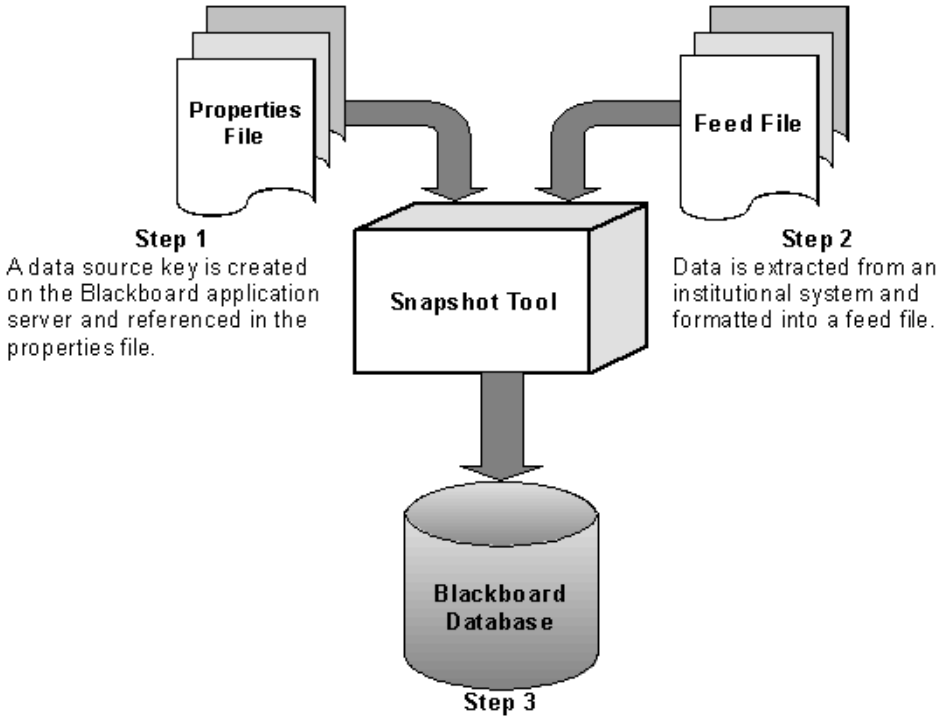
Snapshot can also associate Learning Levels (‘Standards’ in Pre Service Pack 8 versions and ‘Goals’ in Service Pack 8) to courses by using a downloaded Learning Levels CSV (comma separated values) file. The keys in this file are used to associate courses to appropriate Learning Levels at the time the course is created, or in any course modification process executed using Snapshot. The Learning Levels file includes the document identification, the name of the learning level, the name of the standard set the learning level is a part of, the type of standard, and the status of the standard as set by the administrator. To learn more about using CSV files with Snapshot, see Snapshot Syntax.

To populate a Blackboard Learn installation, operations at small institutions and in testing environments can use the Snapshot Command Line Tool manually. Manual use of Snapshot feed files can also be used as a backup when automated systems experience connectivity problems. However, as the Blackboard Learn installation grows, keeping up with data integration manually can rapidly become overwhelming.

Every institution is different so the ways that Snapshot can be automated varies widely, depending on the business rules and the other systems involved. To achieve specific goals, Blackboard recommends thorough planning, including plans to switch Snapshot integrations to the new robust SIS Integration system in Blackboard Learn Service Pack 8. A services engagement with Blackboard Consulting can help plan, scale, and upgrade your integrations.

About the Snapshot Workflow

To push institutional data into Blackboard Learn, Snapshot follows a certain workflow, relying on different types of files that are processed together during a system update. The process diagram illustrates the workflow.



The Data Source Management (DSM) tool is used to create Data Source Keys to associate certain types of information together into the same data source so that it can be processed in one batch. This Data Source Key is added into the Properties File. The Properties File also contains other information necessary to interpret and parse data correctly.

The data in the Feed File has been extracted from an intuitional database and formatted so that it is compatible with the Blackboard database. The Feed File has a header row at the top explaining the types of information contained in the file and individual rows (or one row) containing the data to be modified, divided into columns by a delimiter character.

Planning Frequency of Snapshot Data Feeds

Data can be loaded into the Blackboard Learn database on a schedule determined by the institution and defined by the institution's business rules. The frequency of data feeds can change throughout the year for different data entities.

The following are some of the factors that need to be considered when determining the frequency of data feeds:

- Frequency of adding, modifying, and deleting data in institution systems
- Urgency of the data synchronization
- Institution system performance
- Number of data entities to be fed from institution systems into a Course

Scheduling

To set the frequency, use the Task Scheduler with Windows 2003 or Windows 2000 operating systems and the `cron` command with Linux® or Solaris® operating systems. In addition, Windows users may also use the 'at Scheduler' or the 'at command' from the command prompt.

Note: Frequency can also be controlled with the Snapshot Controller, a custom built utility used to automate processing of one or more Snapshot commands. The Snapshot Controller may be built by your school or provided through Blackboard Consulting.

Setting Up Snapshot Command Line Tool Service

The Snapshot operating parameters are set by configuration properties stored in configuration files. These files are called Properties Files. The Properties Files instructs Snapshot how to interpret and apply the data in the Feed File to the Blackboard Learn database.

While Blackboard supplies a sample Properties File in the snapshot directory (`snapshot.properties`) this file can be renamed and several different versions can be created and used for different processing jobs. The typical items modified in a Properties File are the entry for the Data Source Key, the delimiter used in the feed file, and the list of Blackboard-owned fields.

Snapshot provides different modes depending on the specific task it will be used to perform. This section provides a description of each mode and how to invoke them using the command line syntax.

Configuration Properties Files

The configuration properties files are stored in the `/blackboard/config` directory (`blackboard\config` for Windows users). With Snapshot and the Event Driven API, the following Properties Files are used:

```
service-config-snapshot-soap.properties
service-config-snapshot-jdbc.properties
```

These files are configured at install. If a change needs to be made, backup these files and use them as templates for new config files that include the desired properties.

The following table describes the parameters that can be adjusted.

Parameter	Values
`blackboard.service.log.param.logdef.default.verbosity`	Debug Information Warning Error Fatal
`blackboard.service.log.param.logdef.default.copyToConsole`	True or False

To learn more and additional parameters for property files can, see Using the Properties Files.

Note: All properties files are formatted similar to Windows.ini files, with name/value pairs. Comment lines begin with a hash mark (#).

Invoke Property File for Data Integration

It is necessary to invoke a properties file when running the Event Driven API from a client machine. Invoke this file:

```
service-config-snapshot-soap.properties
```

Database Configuration

The following two files must be changed if the Administrator changes the `bbadmin` password:

```
/blackboard/config/bb-datastores.xml
/blackboard/config/bb-datastores-standalone.xml
```

Initialize Persistence with SOAP

Simple Object Access Protocol (SOAP) is a set of conventions used for invoking code using XML over HTTP. When programming using the Event Driven APIs, the SOAP password (MD5 encrypted) must be passed when run from a client. For example:

```
System.getProperties().setProperty(CIConstants.REMOTE_PASSWORD,"password");
java "-Dremote.access.password=password"
```

Using the Properties Files

Blackboard Learn is installed with a Properties File, located in Snapshot directory and named `snapshot.properties`. This file may be copied, modified and saved under a different name. It is not uncommon to have several different Properties Files for different batch jobs.

How to Edit the Properties File

The Properties File consists of properties and their associated values separated by an "**=**" (equals) sign. Each property is on its own line. Comments are tagged by the "**#**" (pound or hash) sign. For example, the property that defines the delimiter for the Feed File would appear as:

```
# delimeter used for parsing snapshot filesdata.delimeter=|
```

If the institutional data feed file has been formatted with a different delimiter character, for example a "," (comma), the Properties File would have to be modified to match in order to ensure the Feed File data was parsed correctly.

```
# delimeter used for parsing snapshot filesdata.delimeter=,
```

The following table lists all the properties that can exist in a Properties File and their associated values:

Property	Description
announcements.copy	Flag that enables copying of announcements for COPYINTO operations. “Y” or “N” If this is set to “N”, announcements that appear in a source course or organization will not be copied into the destination course. Default value is “Y”.
archive.cs.items	Flag that enables copying of course files for COPYINTO operations. “Y” or “N” If this is set to “N”, links and copies of content that appear in a source course or organization will not be copied into the destination course. Default value is “Y”. If this flag and the course.exact.copy flag are set to "Y", then all course files are included.
assessment.copy	Flag that enables copying of assessments for COPYINTO operations. “Y” or “N” If this is set to “N”, assessments that appear in a source course or organization will not be copied into the destination course. Default value is “Y”.
availability.rule.copy	Flag that enables copying copy adaptive release rules in the course. If set to yes, a COPYINTO operation will copy adaptive release rules that only depend on other copied items.” Y” or “N” Default value is “Y”.
calendar.copy	Flag that enables copying of calendar items for COPYINTO operations. “Y” or “N” If this is set to “N”, calendar items that appear in a source course or Organization will not be copied into the destination course. Default value is “Y”.
cartridge.copy	Flag that enables copying content, assessments, and discussion boards that came from a cartridge. ” Y” or “N” Default value is “Y”.
categories.copy	Flag that enables copying of course or organization categories for COPYINTO operations. “Y” or “N” If this is set to “N”, category membership associated with a source course or Organization will not be copied into the destination course. Default value is “Y”.
chat.archive.copy	Flag that enables copying of chat archives for COPYINTO operations. “Y” or “N” If this is set to “N”, chat archives that appear in a source course or organization will not be copied into the destination course. Default value is “Y”.
chat.session.copy	Flag that enables copying of chat sessions for COPYINTO operations. “Y” or “N” If this is set to “N”, sessions that appear in a source course or organization will not be copied into the destination course. Default value is “Y”.
course.content.copy	Flag that enables copying of course content for COPYINTO operations. “Y” or “N” If this is set to “N”, course content that appears in a source course or organization will not be copied into the destination course. Default value is “Y”.

Property	Description
course.exact.copy	Flag that enables copying of course files for COPYINTO operations. “Y” or “N” If this is set to “N”, an exact copy of a source course or organization will not be copied into the destination course. Default value is “Y”. If this flag and the archive.cs.items flag are set to "Y", then all course files are included.
course.settings.copy	Flag that enables copying of various settings for COPYINTO operations. “Y” or “N” If this is set to “N”, settings from a source course or Organization will not be copied into the destination course, and the destination course will use system default settings. Default value is “Y”.
course.statistics.copy	Flag that enables copying Course Statistics. "Y" or "N" Default value is "Y".
data.delimiter	The column delimiter for the input file. This field is ignored for XML-based feeds. Setting the data delimiter allows for other delimiters to appear in the data. For example, if a comma is used in the data, select another delimiter to prevent errors. By default, the delimiter is a pipe “|”, which is recommended since it is unlikely to appear in most data.
data.source.key	Data source key to be used for the snapshot operation (see Part 2, Data source management, for an overview of datasource keys). The specified data source must first be defined with the DSM Utility before a snapshot or update is run with this key. This property is required and can be overridden from the command line.
discussion.board.archive.copy	Flag that enables copying of Discussion Board archives for COPYINTO operations. “Y” or “N” If this is set to “N”, archives that appear in a source course or organization will not be copied into the destination course. Default value is “Y”.
discussion.board.copy	Flag that enables copying of Discussion Boards for COPYINTO operations. “Y” or “N” If this is set to “N”, discussion boards that appear in a source course or organization will not be copied into the destination course. Default value is “Y”.
drop.box.copy	Flag that enables copying of drop box items for COPYINTO operations. “Y” or “N” If this is set to “N”, drop box items that appear in a source course or organization will not be copied into the destination course. Default value is “Y”.
encrypt.card.number	Flag enabling the system to accept encrypted card numbers (for use with Community Engagement). "N" signifies that the card number is not encrypted. "Y" signifies that the card number is already encrypted. Card numbers are unencrypted using a key shared between the Blackboard Transaction System and the Blackboard Learn. If the card number is not encrypted at import, it will be encrypted by Blackboard Learn using the shared key. Therefore, it is very important that this flag is set correctly; otherwise, card numbers will be processed incorrectly.

Property	Description
encrypt.password	Flag enabling automatic MD5 encryption of passwords provided. “Y” or “N”. The default is “Y”, meaning that the passwords should be provided in plaintext, and the system will automatically encrypt them to the MD5 format used in the database. If the passwords are already being supplied in MD5 format, set this to “N”.
entity.bb.controlled.fields	Attributes that are owned by the Blackboard Learn database. That is, they should not be overwritten by the manual update or snapshot. On insert, the initial value in the feed will be entered into the database. However, on update, the value will not be changed. This is an effective way of updating user information without overwriting user passwords, fax numbers, or other similar information. Entity (as is a variable in the property) is one of person, group, membership, category, or category membership. The value of this property is a comma-delimited list of columns (attributes).
error.delimiter	String used to mark the beginning of an error in the error log file. Default value is (!).
escape.character	Character to be used in escaping the delimiter in use. The default is “/”. For example, if the delimiter is pipe “|” and the escape character is “/”, then the sequence “/|” will indicate a literal pipe character.
glossary.copy	Flag that enables copying the glossary. ” Y” or “N” Default value is “Y”.
gradebook.copy	Flag that enables copying of gradebook content for COPYINTO operations. “Y” or “N” If this is set to “N”, gradebook content that appears in a source course or organization will not be copied into the destination course. Default value is “Y”.
grades.copy	Flag that enables copying the grades in the Course. ” Y” or “N” Default value is “Y”.
groups.copy	Flag that enables copying of course group content for COPYINTO operations. “Y” or “N” If this is set to “N”, course group content that appears in a source course or organization will not be copied into the destination course. Default value is “Y”.
header.validation	Flag enabling validation of header information. Determines whether labels provided in the first line of the feed file are supported for the current operation. “Y” or “N”
log.stdout	Flag to indicate the display of status information to the system console. “Y” or “N” The default value is “N”.
max.error.count	Maximum number of errors allowed before processing of the current file terminates. Records that have been successfully processed are not rolled back. A setting of “-1” will allow an infinite number of errors during processing. A setting of “0” will terminate the process at the first error.

Property	Description
membership.copy	Flag that enables copying of membership information for COPYINTO operations. “Y” or “N” If this is set to “N”, student enrollments and staff students that appear in a source course or organization will not be copied into the destination course. Default value is “Y”.
membership.datasource.key	The data source key to be assigned to enrollments of a course that is created through Course Copy. If no data source key is identified, the enrollments will use the same data source key as the source. This parameter is only used when “membership.copy” is set to “Y”.
membership.exact.copy	Flag that enables copying private membership data. ” Y” or “N” Default value is “Y”.
parse.allow.default	Enables the administrator to allow incorrect data in the snapshot feed to be reset to the default. “Y” or “N”. Default is “N”
reconcile	Flag that, when set to “Y”, enables a Course Copy operation to completely replace the settings and Course Menu of the destination course. This works even when copying into a new course (removes the default settings and default Course Menu and replaces it with the source course). When the flag is set to “N” the destination vourse will contain Course Menu items from both the original destination course and the source course. Note that setting this to "Y" can result in the loss of data in the destination course.
snapshot.batch.size	Flag designating the number of records handled in a database transaction. Default is set to 300.
staff.information.copy	Flag that enables copying of staff information for COPYINTO operations. "Y" or "N" If this is set to "N", staff information that appears in a source course or organization will not be copied into the destination Course. Default value is "Y".
suppress.clone.events	Flag that will suppress CLONE events. If set to yes, CLONE events will be suppressed. ” Y” or “N” Default value is “Y”.
tasks.copy	Flag that enables copying of tasks for COPYINTO operations. “Y” or “N” If this is set to “N”, tasks that appear in a source course or organization will not be copied into the destination course. Default value is “Y”.
wait.length	Used to configure pause between persistence actions. (seconds) The default value is”-1” indicating no waiting This value can be set to improve database performance (reducing the impact of the Snapshot operation on the system), at the expense of total execution time.

Dependencies

If `membership.exact.copy` is set to Y, then `membership.copy` will be forced to Y. Data dependent on enrollments can only be copied if the enrollment data itself is being copied.

If `grades.copy` is set to Y, then `membership.exact.copy` will be forced to Y. Data dependent on enrollments can only be copied if the enrollment data itself is being copied.

About Feed Files

The data that is fed into Blackboard Learn to create users, courses, update records and so on exists in a “feed file”. Snapshot Feed Files can be character-delimited flat files or XML files that conform to IMS standards. Blackboard adheres to the Global Community XML encoding standards, which commonly uses UTF–8.

To accept Multi-byte information, files must be in Unicode format. All Unicode formatted files must include the Byte Order Mark (BOM). If a BOM is not specified, the system reads it as ISO-8859-1. The supported formats are as follows:

- UTF-8
- UTF-16LE
- UTF-16BE
- ISO-8859-1

ASCII is also a supported format, since it is a subset of ISO-8859-1.

Using Delimiters and Escape Characters

For data to be parsed correctly from a Feed File, Snapshot needs to know where the data for each different field ends. A character that does not regularly appear in the data is usually selected as the “delimiter” – the character that separates one field from another in a row. The “|” pipe character is the default delimiter character in the Properties File.

Occasionally the delimiter character will appear in the data. Unless the system knows that this particular instance of the delimiter character is actually part of the data itself and not denoting the division of a field, the data feed will contain errors or fail all together. In order to tell the system that a particular delimiter character is part of the data set it must be “escaped”.

To be able to escape a delimiter, an escape character is defined in the Properties File. The “/” character is the default escape character.

How to Change the Delimiter

To change the default delimiter character, edit the Properties File. Any SQL character may be used as a delimiter. Find the line in the Properties File that reads:

```
# delimiter used for parsing snapshot files
data.delimiter=|
```

Change the character on the right side of the equals “=” sign to the desired character. If this character appears in a data field, it must be escaped in every field where it occurs. For this reason, selecting a character that might commonly appear in the data fields is not recommended.

How to Change the Escape Character

To change the default escape character, edit the Properties File. Find the line in the Properties File that reads:

```
# Determines character used to escape delimiter

escape.character=/
```

Change the character on the right side of the equals "=" sign to the desired character. This character would precede any delimiter character that appears in all data fields. For example, if the hyphen "-" has been set as the delimiter, every time a hyphen appeared in a data field it would be escaped by placing the "/" character before it. In the example below, the last name is Smith-Jones, so the hyphen in the last name must be escaped like this:

```
EXTERNAL_PERSON_KEY-USER_ID-SYSTEM_ROLE-FIRSTNAME-LASTNAME-EMAIL-
INSTITUTION_ROLE

2000-jsmithjones100-none-John-Smith/-Jones-jsmith@nothing.com-Student
```

Using the Snapshot XML Feed File Format

Compared to a flat file feed, an XML Snapshot file has increased flexibility in that the XML files can process one data type at a time, or combine multiple data types to be processed at once. While Blackboard Learn does not validate against the XML Data Dictionary defined in Data Format Tables for XML Files, the code must be well formed, meaning each nested tag must be properly closed.

The Snapshot Tool will process IMS compliant XML files, version 1.01. For information about IMS standards for XML files, view the following Web page on the IMS site:

http://www.imsproject.org/enterprise/enbind03.html

Since entities in the XML file are typed, there are no specific requirements to separate entities into separate Feed Files. However, the action and data source attributes specified in the IMS standard are ignored. In processing the IMS XML format, the Blackboard Learn Snapshot Tool still requires the same logic and workflow as used with a flat file.

XML Template Tags

The following refer to specific tags in the XML template:

- Extension tags: These tags are not supported in the IMS specifications; they are only supported by Blackboard Learn. As they are adopted by the IMS project, Blackboard will phase them into proper XML definition.
- Categories and Category Memberships: These are not IMS data types and are not supported by IMS; they are supported by Blackboard Learn.

The following tips are helpful for troubleshooting Snapshot XML files:

- If a user fails to be created in an XML file then their membership will also fail. Many aspects of the XML file are related, therefore is one area fails, it may cause others to also fail.
- Blackboard recommends using an XML viewer when viewing the .invalid file (error log). This file should be referenced if an error occurs.

XML Feed File Example

An example of an XML Feed File can be view in Data Format Tables for XML Files.

Using the Snapshot Flat File Format

The Snapshot Feed File can be a “flat file”, another term for a delimited text file. A Feed File may encompass information about any of the ten entities that are transferred to Blackboard Learn. A Feed File consists of a header row that identifies each of the columns or fields of data followed by one or more rows of data records.

Header Row

The Header Row in a Feed File contains Field Names, which are the exact names of the fields included in the file. These names are listed in the tables in the feed file topics, starting with Category Data Feed Elements. The header fields are delimited with the same character as the data records.

Fields can appear in the header row in any order, but fields not recognized by Blackboard Learn cannot be included or errors will occur or the operation will fail completely.

Note: Snapshot feed files work in both Right to Left and Left to Right operating system environments. The order of the fields in the header row is not important, only that it matches the order of the data rows. The delimiter for Snapshot feed files used in a Right to Left environment must be a Tab. The commonly used pipe (|) can cause some fields to reverse order, misaligning the header row with the data rows.

Each Feed File type has a minimum number of required fields that need to be identified in the header row.

The minimum required fields for a user Feed File are:

- EXTERNAL_PERSON_KEY
- USER_ID
- SYSTEM_ROLE
- FIRSTNAME
- LASTNAME
- EMAIL
- INSTITUTION_ROLE

The minimum required fields for a Course or Organization Feed File are:

- COURSE_ID
- EXTERNAL_COURSE_KEY
- COURSE_NAME

The minimum required fields for an enrollment or staff Students Feed File are:

- EXTERNAL_COURSE_KEY
- EXTERNAL_PERSON_KEY
- ROLE

An example of a header row for a user Feed File:

```
EXTERNAL_PERSON_KEY|USER_ID|SYSTEM_ROLE|FIRSTNAME|LASTNAME|EMAIL|INSTITUTION_
ROLE
```

Data Rows

After the header row, data records, one per line are added to the Feed File. Records are separated by a carriage return, a linefeed or the combination of the two. The record separator is not configurable.

Some data feed elements have sets of string constants for their possible values. For example, the attribute Institution_Role maps to coded characters in the database.

Information that is not given (left blank) for fields that are included in the header is not changed during the operation. Exceptions:

1. If the password field is null the User ID will be set as the password. To clear a field, enter a single space.
2. To clear the email field, enter a single @ symbol, as it is a requirement. If the email address field is left blank, the following error will be shown "(!)Invalid value provided: Email. Review documentation".

Any fields in the Feed File not found in the feed file topics, starting with Category Data Feed Elements, will be ignored.

Data contained in each record must conform to values that are acceptable to Blackboard Learn. For acceptable values for each field, see the feed file topics, starting with Category Data Feed Elements.

Note: The delimiter for all Snapshot feed files to be used in a Right to Left environment must be a Tab. The commonly used pipe (|) delimiter can cause some fields to reverse order. The order of the data fields in the data rows must match the order of the fields in the header row. Additionally, any data that is entered as mixed orientation, some text formatted left to right and some formatted right to left in the same file will cause data errors or cause the operation to fail.

Example of Data Rows in a Delimited User Feed File

```
EXTERNAL_PERSON_KEY|USER_ID|SYSTEM_ROLE|FIRSTNAME|LASTNAME|EMAIL|INSTITUTION_
ROLE
2000|jsmith001|none|John|Smith1|jsmith@sununiversity.edu|Student
2001|kthomas001|none|Kyle|Thomas2|kthomas@sununiversity.edu|Student
2002|ttsai001|none|Tevis|Tsai|ttsai@sununiversity.edu|Student
2003|lgonzales001|none|Lois|Gonzales|lgonzales@sununiversity.edu|Student
2004|mmacneil001|none|Megan|MacNeil|mmacneil@sununiversity.edu|Student
2005|mmacneil002|none|Mark|MacNeil|mmacneil1@sununiversity.edu|Student
2006|ddishez001|none|Dan|Dishez|ddishez@sununiversity.edu|Student
```

```
2007|rrondelle001|none|Ronda|Rondelle|rrondella@sununiversity.edu|Student
2008|jsmith002|none|John|Smith2|jsmith1@sununiversity.edu|Student
```

Delete Records File

A “Delete Records” file will be the file format used to purge unwanted data from Blackboard Learn and may encompass information about any of the entities that are to be transferred to the Blackboard database. The file need only contain certain key fields in order to delete data records from Blackboard Learn. For a list of the required fields for each entity type, see the feed file topics, starting with Category Data Feed Elements. The file format for the Delete Records file is as follows:

Header: Field Names. The exact names of the fields included in the file.

Example (Membership)

```
EXTERNAL_COURSE_KEY|EXTERNAL_PERSON_KEY
Math101.1.Fall1999|1074202|course_builder
History176.6.Spring2000|324-765-0098|Instructor
Chem401.1.Fall1999|uberk1278|Student
```

Invalid Data

Incorrectly formatted data is handled as follows:

- Strings that exceed the specified max length for that column will be truncated. Data Source Keys that exceed the specified max length for that column will cause the record to fail.
- Data types that cannot be converted (such as dates) are ignored. (They are treated as a space or as missing.)
- Invalid column headers are ignored.

Note: Unsupported feed elements will not cause the feed to fail—they are ignored to preserve backward compatibility with older feed generators.

Snapshot Modes

Depending on the operation specified in the command line processed by Snapshot, the records specified in the Feed File will be affected differently. The four primary modes of snapshot are:

- Snapshot Mode
- Manual Mode
- Remove Mode
- Course Copy (COPYINTO)

Snapshot Mode

The Snapshot Mode processes a Snapshot against all records in the database for the specified data source key. Records are processed in the following ways:

- If the record is in the Feed File, but not in the Blackboard Learn database, an insert is performed and the record is added.
- If the record is in the Feed File and in the Blackboard database, an update is performed to Blackboard Learn database.
- If the record is not in the Feed File, but in the Blackboard database, the record is disabled. If a later Snapshot contains a record in the disabled state, the record is enabled and updated. The entity may only be re-enabled by including it in a subsequent Snapshot file, not through the Blackboard user interface. To learn more see the topic, Data Lifecycle.

Note: Because all entities within a given data source are disabled if not included in a feed file, it is important to double-check the contents of the file before running in snapshot mode. An improper feed file could result in large amounts of data being disabled. For small updates and testing, use manual mode instead.

Manual Mode

In Manual Mode, only the specific records in the feed file are processed. Other records in the data source key are not affected. Records are processed in the following ways:

- If the record is in the Feed File, but not in the Blackboard database, an insert is performed and the record is added.
- If the record is in the Feed File and in the Blackboard database, an update is performed to Blackboard Learn database.
- If the record is not in the Feed File, but in the Blackboard database, the database record is not altered.

Because Manual Mode does not affect records not listed in the feed file, users may use Manual Mode to add a small number of records without worrying about the possibility of disabling records.

Remove Mode

Remove Mode is used to remove courses, enrollments, staff students, and so on. Like Manual Mode it only processes those records contained in the Feed File. Data is purged for each record identified in the Feed File. This is a permanent delete. The record is permanently removed from the database, including any associations of the record (that is, if the user record is removed, any enrollments and files associated with the user are permanently deleted. If a course is removed, any enrollments associated with the course are also deleted).

Course Copy (COPYINTO)

Course Copy Mode, called “COPYINTO," allows automatic management of course and organization content. The commands can be used to add content from Course Templates to many different courses or organizations. The COPYINTO command can be issued as part of a Snapshot or Manual operation, or as a stand-alone operation by invoking the `CRS_COPYINTO` or `ORG_COPYINTO` command.

To perform a Snapshot or Manual operation using COPYINTO, use `CRS_SNPSHT` or `CRS_MANUAL` with the `-t` option using the `TEMPLATE_COURSE_KEY` field in the feed file. To copy both content, and links use `"-a true`"; by default the setting is `"-a false`" which means if the `-a` option is omitted then the links will not be copied.

Using the COPYINTO in this way is only meant for populating new courses. Using this method with existing courses will not copy the data. To use COPYINTO with existing Courses, use `CRS_COPYINTO` instead.

Note: In instances where a copy occurs on update, if the information from the source has already been added to the destination course, it may be added again resulting in duplicate data. See the description for the "reconcile" snapshot properties option.

COPYINTO operations use the `Template_Course_Key` or `Template_Organization_Key` to specify the source of the copy operation. Depending on the ownership setting for `Template_Course_Key` or `Template_Organization_Key`, the COPYINTO operation will process the information according to the following table.

Operation	Ownership of Template Key	Result
COPYINTO update	Blackboard	Copy
not Blackboard	Copy	
COPYINTO insert	Blackboard	Copy
not Blackboard	Copy	
Snapshot update	Blackboard	No Copy

Operation	Ownership of Template Key	Result
not Blackboard	Copy	
Snapshot insert	Blackboard	Copy
not Blackboard	Copy	
Manual update	Blackboard	No Copy
not Blackboard	Copy	
Manual insert	Blackboard	Copy
not Blackboard	Copy	

Update Logic

When a record is updated, ownership of the data fields, if applicable, is processed so that data owned by Blackboard Learn is not changed. Null values in a data field are ignored and the data is not changed. Therefore, a field is only updated if it is non-blank and not owned by Blackboard Learn. To clear a field (for example to remove a middle name), set it to a single space character.

Snapshot Command Line Syntax

Commands issued to Snapshot must follow a specific syntax to be processed correctly. Flags are used to specify arguments in a Snapshot command line. Snapshot is located in the `blackboard\apps\snapshot\bin directory`.

Snapshot Flags

Snapshot commands consist of the following components:

- The "-f" flag and the invocation (command) that is to be run
- The "-t" flag and the full path location to the Feed File
- The "-C" flag and the full path location to the Properties File
- The "-V" flag and the fully-qualified domain name of the virtual host that will be used

Flags are case sensitive, so an upper case "C" needs to be used to denote the full path name to the Properties File and an upper case "V" needs to be used to denote the virtual host name. The "f" and "t" flags are in lower case.

Example

The syntax for an entire Snapshot command will look something like the following.

Windows

```
Snapshot –f [action] –t [x:\location of feed file] –C [x:\location of
properties file] –V [fully-qualified server name])
```

UNIX

```
./snapshot –f [action] –t [/location of feed file] –C [/location of
properties file] –V [fully-qualified server name])
```

Taking the above criteria, a sample Windows syntax can be built like this:

```
Snapshot –f CRS_MANUAL –t c:\snapshotstuff\feedfiles\fallCourses.txt –C
c:\snapshotstuff\snapshot.properties –V blackboardserver.acme.edu
```

Taking the above criteria, a sample UNIX syntax can be built like this:

```
./snapshot.sh –f CRS_MANUAL –t /snapshotstuff/feedfiles/fallCourses.txt –C
/snapshotstuff/snapshot.properties –V blackboardserver.acme.edu
```

Snapshot Command Line Examples

Snapshot can be used to update the Blackboard database in many ways based on an institution's business rules. The following sections contain use cases with a brief description of the process used to execute them.

For all of the use cases, "webserver host" indicates that users should insert the name of their webserver host, for example `blackboard.school.edu`.

Use Case 1

Organize students in the Information Systems Management program into two groups: Project Management (PM) and ISD (Information System Development).

1. Create two Data Source Keys: `IFSM.PM.02` and `IFSM.ISD.02`.
2. Associate the Data Source Key to the selected user group in the `snapshot.properties` file. Insert the users with the desired Snapshot command:

   Controls whether to copy user grades during COPYINTO operations. This will also copy enrollments. The available options are Y/N.

   ```
   groups.copy=Y
   ```

   Designates flag to enable membership data copying for configuration property `filemembership.copy=Y`.

   Windows

   ```
   C:\blackboard\apps\snapshot\bin>snapshot –V "webserver host" -f
   usr_snpsht –t ..\Demo\users\ISD.02.csv -C
   ..\data\snpshot.properties
   ```

   UNIX

   ```
   /usr/local/blackboard/apps/snapshot/bin >./snapshot.sh –V "webserver
   host" -f usr_snpsht –t ../Demo/users/ISD.02.csv -C
   ../data/snpshot.properties
   ```

3. Run the same process again using Snapshot command with the file `PM.02.csv` associated with the key `IFSM.PM.02`.

Use Case 2

A student is disabled in the system until their financial aid arrives. Disabling a record rather then deleting it is often the best option if the record will be activated at a later time. When a user is disabled the user record exists within the database but is not active in Blackboard Learn. The user cannot login. However, the user record appears in the System Control Panel in gray text with an icon that signals its disabled status.

A student can be disabled by changing an element in the student record and running the Snapshot command again using the same source key.

Use Case 3

Organize a Statistics course so that it is offered in two sections over the Summer 2006 semester.

1. Create Data Source Keys for the Course with two sections, offered in a Summer 2006 semester:

Windows

```
C:\blackboard\apps\snapshot\bin\dsm -V"webserver host"-f create -b
stats.101.SUM.10 -d Statistics 101 - section 10 summer 2002"

C:\blackboard\apps\snapshot\bin\dsm -V "webserver host" -f create -b
stats.101.SUM.11 -d "Statistics 101 - section 11 summer 2002"
```

UNIX

```
/usr/local/blackboard/apps/snapshot/bin/dsm.sh -V "webserver host" -f
create -b stats.101.SUM.10 -d "Statistics 101 - section 10 summer 2002"

/usr/local/blackboard/apps/snapshot/bin/dsm.sh -V "webserver host" -f
create -b stats.101.SUM.11 -d "Statistics 101 - section 11 summer 2002"
```

2. Set the properties to the desired key each time a Course is inserted.
3. Insert the first and second sections of the Statistics course:

Windows

```
C:\blackboard\apps\snapshot\bin>snapshot -V "webserver host" -f
crs_snpsht =t ..\demo\course\stat_sun_02_10.csv -
C ..\data\snapshot.properties

C:\blackboard\apps\snapshot\bin>snapshot -V "webserver host" -f
crs_snpsht =t ..\demo\courses\stat_sun_02_11.csv -
C ..\data\snapshot.properties
```

UNIX

```
/usr/local/blackboard/apps/snapshot/bin >./snapshot.sh -V "webserver
host" -f crs_snpsht =t ../demo/courses/stat_sun_02_10.csv -
C ../data/snapshot.properties

/usr/local/blackboard/apps/snapshot/bin >./snapshot.sh -V "webserver
host" -f crs_snpsht =t ../demo/courses/stat_sun_02_11.csv -
C ../data/snapshot.properties
```

Use Case 4

Enroll students in a course. The file used to enroll students into a course uses the same idea as a relational database. The file to enroll users is created by combining external_person_key and external_course_key.

1. Create a Data Source Key for the course enrollment:

   Windows

   ```
   C:\blackboard\apps\snapshot\bin>dsm -V "webserver host" -f create -b
   enr.stat.10.sum -d ">enrollment for Statistics 101 section 10 for
   summer of 2002 semester"
   ```

   UNIX

   ```
   /usr/local/blackboard/apps/snapshot/bin/dsm.sh -V "webserver host" -f
   create -b enr.stat.10.sum -d "enrollment for Statistics 101 section 10
   for summer of 2002 semester"
   ```

2. Set this key in the `snapshot.properties` file before running the Snapshot command.
3. Enroll selected Students into the course.

Snapshot Override Syntax

Snapshot Override is a special mode of Snapshot that has its own format to allow the Data Source Key to be specified from the command line rather than in the Properties File. This can save time by allowing for a quick update without having to create a new properties file.

Shapshot Override Flags

Snapshot Override commands consist of the following components:

- The “-D” flag and the Data Source Key that will be used to override the current one in the Properties File.
- The “-f” flag and the invocation (command) that is to be run.
- The “-t” flag and the full path location to the Feed File.
- The “-C” flag and the full path location to the Properties File.
- The “-V” flag and the fully-qualified domain name of the virtual host that will be used.

Flags are case sensitive, so an upper case “D” needs to be used to denote the Data Source Key, an upper case “C” needs to be used to denote the full path name to the Properties File and an upper case “V” needs to be used to denote the virtual host name. The “f” and “t” flags are in lower case.

Example

The syntax for an entire Snapshot Override command will look something like this:

```
Snapshot_override "-Ddata.source.key=[key] –f [Snapshot Action] –t [feed
file] –C [properties file] –V [fully-qualified server name]
```

Taking the above criteria, a sample Snapshot Override syntax can be built like this:

```
Snapshot_override "-Ddata.source.key=fall_COURSES_05" –f CRS_SNAPSHOT –t
c:\snapshotstuff\feedfiles\fallCourses.csv –C
c:\snapshotstuff\snapshot.properties –V "blackboardserver.acme.edu"
```

Setting Up SOAP for Use with the API

SOAP (Simple Object Access Protocol) is an XML format that handles the execution of server side code remotely, using the Web. Blackboard Learn supports SOAP implementation of Loaders and Persisters in the Event Driven APIs. SOAP is activated through a change to the configuration. When this change is made, the DSM Tool and Snapshot will both use SOAP.

Configuration Change

The DSM Tool and Snapshot share an environment file, which can be altered to use SOAP. This file is located at `Blackboard/apps/snapshot/config/env.cmd`.

By default the config file is configured directly to the database as:

```
CONFIG_FILE=%bbdir%\config\service-config-snapshot-jdbc.properties
```

To implement SOAP, change the property to read:

```
CONFIG_FILE=%bbdir%\config\service-config-snapshot-soap.properties
```

Command Line Syntax

When running the invocation through SOAP, the integration user account password information must be added as shown. "Script" is either dsm or snapshot:

```
script -P "password" [additional flags]
```

Using an SSL Connection

Administrators may choose to use an SSL connection for SOAP Snapshots. The following instructions can be used for both Windows and UNIX systems.

Locate the SSL Certificate

Check that the server SSL certificate is in X.509 form. If not, export the server's SSL certificate to a file. On Windows, use the Certificate Export Wizard to export the certificate. On Unix, the certificate is located at the path set in `/usr/local/blackboard/apps/httpd/conf/ssl.conf`.

Import the Server SSL Certificate to the Cacerts Keystore on the Client

The cacerts keystore is located in the Java JDK installation under `jre/lib/security/cacerts`. Use the keytool to import the server certificate, for example:

```
keytool -import -file /usr/local/blackboard/apps/httpd/conf/certs/server.crt
-keystore /usr/java/jdk1.6.0_18/jre/lib/security/cacerts -trustcacerts
```

The Administrator will be prompted for a keystore password. The default keystore password for cacerts is “changeit”. The Administrator may change that password using the keytool.

The server certificate may also be imported to a different keystore, but the Administrator should check the root Certificate Authority (CA) certificate from the issuing Certificate Authority exists in the cacerts keystore.

The above are simplified instructions. For more details on importing SSL certificates and working with the Java keystore, please consult the "keytool" documentation on Sun's web site. Choose the appropriate link for your platform:

UNIX: http://java.sun.com/javase/6/docs/technotes/tools/solaris/keytool.html

Windows: http://java.sun.com/javase/6/docs/technotes/tools/windows/keytool.html

Edit the service-config-snapshot-soap.properties File

In this file, change the following key values:

```
blackboard.service.soap.param.encrypt=true
blackboard.service.soap.param.truststore=location of keystore
```

where `location of keystore` is replaced by the path to the keystore (see Step 2).

Initialize Persistence with SOAP

Simple Object Access Protocol (SOAP) is a set of conventions used for invoking code using XML over HTTP. When programming using the Event Driven APIs, the SOAP password (MD5 encrypted) must be passed when run from a client.

Example

```
System.getProperties().setProperty(CIConstants.REMOTE_PASSWORD,"password");
java "-Dremote.access.password=password"
```

Data Mapping

To integrate institutional systems data with Blackboard Learn institutional data attributes must be mapped to attributes in the Blackboard Learn Data Dictionary. The Blackboard Learn Data Dictionary is presented in a series of tables that begin with Category Data Feed Elements.

Blackboard Learn Attributes

Blackboard Learn has a minimal set of required attributes (fields) for records of each entity type. This minimal set of attributes provides the basic integration capability. By using additional attributes available in Blackboard Learn, more complex integration can be performed for a richer and more convenient online learning experience.

Ownership Information

Only authorized personnel should have access to the systems to perform changes. An "ownership" flag determines whether data is owned (controlled) by the institutional system, or whether changes are allowed through the Blackboard Learn user interface. Designating an attribute as owned by the institutional system will override any changes to that attribute made through the user interface. For example, if a user were to change their password through the UI, the next time data was transferred from the institutional system, the password attribute would be overwritten if that attribute were owned by the institutional system. Conversely, if the data is owned by a Course, a change from the institutional system will not override the change in Blackboard Learn.

Note: Institution systems will not contain data corresponding to every attribute in Blackboard Learn. Data that appears only in a Course should not be owned by the institution system.

Data in Multiple Enterprise Systems

For data records that appear in multiple institution systems, a decision must be made as to which system's records will override the other system's records during data integration, ensuring the newest or most relevant data is in the Blackboard database. For example if data is integrated from a Student Information System (SIS) as well as data from an Alumni database, one system key must take precedence over the other. This can be accomplished in two different ways:

Develop logic that filters out non-authoritative data as the feed files are created.

-or-

Enter all the data into a Snapshot feed file but make sure the authoritative data is entered last.